intel®

ARTICLE
REPRINT

AR-486

June 1990

Flash Memory Outshines ROM and EPROM

SAUL ZALES

INTEL CORPORATION

Order Number 295050-001

Flash Memory Outshines ROM and EPROM

SAUL ZALES, INTEL CORP., FOLSOM, CALIF.

Flash memories may improve product costs and reliability in many markets

As competitive pressures continue to mount, project and design leaders find themselves taking on more responsibility for overall system costs. In many cases, systems designers are expected to design with an eye toward controlling costs in all phases of the product life cycle, from conception through design and manufacturing—even up to post-sales service.

Beyond paying strict attention to total life-cycle costs, today's systems designers face other significant challenges. These include choosing the most efficient CPU architecture, designing for cross-vendor system connectivity, maintaining component and system quality, and building systems that can be serviced easily. Software issues facing project and design leaders include planning for past and future compatibility, minimizing code size, and balancing system performance and stability with time-to-market concerns.

Memory designs have coalesced around basic choices: disk and DRAM architectures, and EPROM-based designs. Although these technologies have been improved over the years, they still require designers to make some trade-offs for certain applications. One such application that has taken on a growing importance for many manufacturers is embedded control.

In the past decade, designers of electromechanical systems have come to rely heavily on the use of embedded microcontrollers. These parts have vastly improved control functionality and performance. Consider, for instance, the growing number of consumer-electronics items that are microprocessor controlled: microwave ovens, washing machines, VCRs, audio systems, and exercise equipment, to name but a few. These products ship with operating code stored in ROM or EPROM; the manufacturer assumes that the code stored in these memories will never change.

Consumer products are not the only items equipped with embedded microcontrollers, of course. Industrial machines, office-automation equipment, medical equip-

ment, communications equipment, avionics systems, and data loggers all include code stored in ROM or EPROM. In nonconsumer products, code changes are more likely to occur, requiring frequent memory fixes. Reasons for such changes include evolving customer needs, frequent demands for new features, improved algorithms, changing connectivity protocols, and eliminating software bugs.

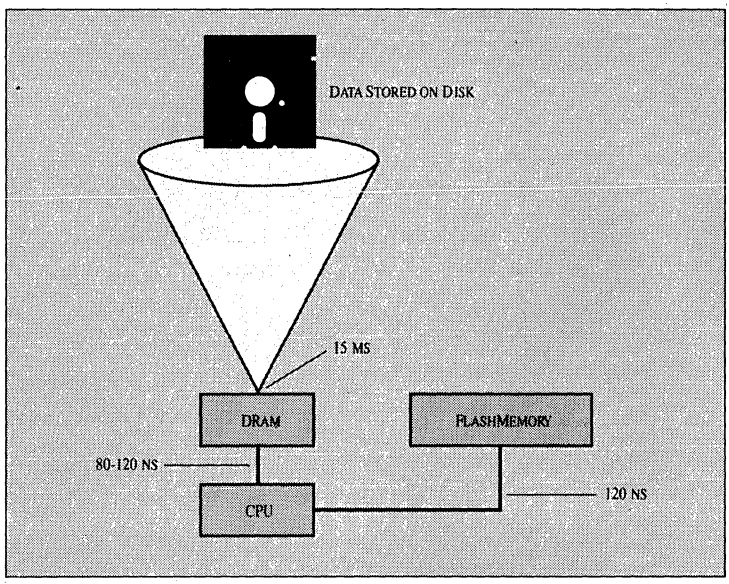
Code updates are impossible with some memory technologies; with others, they incur high costs and threaten product reliability (see box, "Memory Alternatives: Major Trade-Offs"). Designers of products that are likely to require code updating in the course of their life cycles should look beyond conventional memory options and investigate a technology that is well suited to such applications: reprogrammable flash memory.

Reprogrammable flash memories have the potential to improve product costs and reliability in many market segments. For example, ROM and EPROM parts are currently used in electronic engine controllers for automobiles. These parts cannot be replaced; they are sealed under a moisture-resistant coating. To change the code enclosed in one EPROM, the service center must replace the entire controller module, which costs about $200.

Starting in 1993, Delco and a few automakers will begin using flash memory in these modules. With flash memory in place, if a code change is required—for instance, if the Environmental Protection Agency mandates a code change to reduce exhaust emissions—the service center can simply reprogram the memory using a serial link from the service bay's diagnostic computer. No parts need to be replaced, eliminating the risk of damage to other components in the module.

Although flash-memory parts are more expensive than EPROM components, the additional up-front costs eliminate expenses that occur later in the product's life cycle—that is, when code has to be updated. Although customers generally assume the burden of paying for code upgrades, there are certain hidden costs to manufacturers that, taken together, can be significant. For example, consider the cost difference between transferring an upgrade electronically—as can be done with flash memory—and sending a highly paid technician to make a service call.

For whatever reason, most manufacturers do not include upgrade costs in their overall system cost estimates. If a year or two after a product is sold a code update is required, that expense usually is unaccounted for in the system's overall cost. But just because the expense is unaccounted for doesn't mean money isn't spent. Costly updates to existing products will show up on the organization's bottom line somehow. For this reason, even if only one code change is anticipated in the lifetime of a product, designers should opt for flash memory.



In a typical computer, the CPU's performance is slowed by the data bottleneck created by relatively sluggish disk access times. Embedding code in flash memory can eliminate this bottleneck for key operations.

■ PC APPLICATIONS

Flash memory fits extremely well in the embedded-control world, but its usefulness is not limited to such applications. It is becoming more apparent that flash memory will play a critical role in the reprogrammable environment as well. For example, today's personal computers are based on DRAM and disk drives. Obviously, these systems provide acceptable performance to users. Their performance can be improved more, however, by employing firmware based on flash memory.

ONE AREA IN WHICH FLASH MEMORY COULD BOOST PC PERFORMANCE IS IN THE COMPUTER'S BIOS

One area in which flash-memory firmware could greatly boost PC performance is in the computer's basic input/output system (BIOS). A PC's BIOS contains the system initialization, a power-on self-test, and basic component-level drivers. Currently, a computer's BIOS typically is stored in ROM or EPROM, which means that, in essence, it is an embedded-control memory. Without the nonvolatility these memories offer, the system could not initialize itself sufficiently to load software from the disk. This is because the CPU cannot read directly from disk—the disk's access time is much too slow.

Although the average PC user does not think about changing a machine's BIOS, a number of major PC vendors are planning to use flash memory for storing BIOS for several reasons. One is the rapidly changing nature of microcomputer technology. Consider the increase in system complexity from the PCs of 10 years ago to today's top-of-the-line machines. Today's 32-bit PCs offer the computing power of minicomputers. Computer makers need technology that allows for rapid adaptation to ever-changing situations.

The open nature of microcomputer systems adds to the need for flexible BIOS. Multiple vendors develop products that rely on the system's BIOS. In other words, the BIOS drivers hold the key to compatibility with both older hardware and software and newer products. The average stand-alone user who buys a computer, adds in a couple of boards, and runs a half dozen or so popular software packages usually doesn't need to worry about BIOS compatibility. But what about MIS or DP managers at Fortune 500 organizations? They may be responsible for hundreds or

even thousands of PCs fitted with any number of different add-in boards and running hundreds of different software packages. Incompatibilities are likely to abound in this type of environment.

With BIOS stored in EPROM or ROM devices, revisions to BIOS code are impractical. The end result is that many users must make do with systems that do not act predictably with certain software or hardware. If BIOS is stored in flash memory, however, vendors can provide a disk with new code and a simple upgrade utility.

Designing a flash-memory-based BIOS poses similar considerations as designing for the embedded-control environment (see box, "Designing with Flash Memory"). The system must contain a 12-V power supply regulated to ±5 percent. In addition, designers must deal with issues regarding the boot code. If power goes down midway through the BIOS upgrade, from where will the system boot? A boot PROM with the basic hardware initialization code could be included for safety's sake. It could be shadowed out once the flash memory has been properly initialized. Still, including a boot PROM incurs added costs and uses board space.

Designers at Ing. C. Olivetti have built a flash-memory BIOS without the PROM safety net. The Italian company's new 80486-based microcomputer includes a flash-memory BIOS without a boot PROM. Designers decided that the risks involved in eliminating the PROM were minimal. A typical flash-memory upgrade takes 7 to 10 seconds. The chances of power going down during those 7 to 10 seconds are not great enough to merit inclusion of the PROM, Olivetti decided.

■ BEYOND BIOS

BIOS is only one PC component that can benefit from the flexible-firmware concept. The setup and diagnostics programs

WITH FLASH MEMORY, IF OPERATING SYSTEMS ARE UPGRADED, USERS CAN SIMPLY REPROGRAM EMBEDDED PARTS

that ship with every system can be stored in flash memory. Another design improvement is to put the operating system into flash memory. Consider the PC's boot sequence. When the system is powered up, the CPU executes the BIOS hardware initialization, performs a power-on self-test, and spins the disk up to speed. Only after the disk stabilizes at its operating velocity can the CPU begin to read the operating system.

Most operating-system code is read-only, which means that, theoretically, it could be included in the BIOS ROM to allow for faster system power up. The reason that this has not been done on a wide scale is simple: Systems designers recognize that operating systems evolve and improve over time, and that a flexible environment is required. With flash memory, however, concerns about upgrades are eliminated. If operating systems are upgraded, users can simply reprogram flash memories, as with flash-memory BIOS. Two major operating-systems houses, Microsoft Corp. and Digital Research Inc., now offer their operating systems in a form suitable for storage on ROM or flash memory.

In addition to storing the operating system in desktop or laptop systems, flash memory can increase network performance for intelligent terminals and diskless engineering workstations. Large networks, such as those used in airline-reservation systems or retail point-of-sale systems, often bog down during peak transaction periods. The load on the network can be reduced by storing the operating system, LAN protocols, or even scheduling or pricing tables in flash memory. Updates can occur during off hours through the network itself. Curtis Inc. (St. Paul, Minn.), which has offered EPROM or SRAM solutions to this problem for a number of years, now also offers flash-memory products to improve network performance.

■ OFFICE APPLICATIONS

Firmware based on flash memory can lead to significant efficiency improvements in several office applications. For instance, most offices now deploy laser printers for high-quality output. Laser printers allow users to produce typeset-quality memos, presentations, and the like. In many cases, the type fonts used with laser printers are stored in software on a PC's hard-disk drive. Users download these fonts as they are needed to the printer's RAM. Every time a font is changed, the downloading process must take place. Alternatively, many laser printers are equipped to take fonts from ROM- or EPROM-based cartridges that plug directly into the printer. The problem with ROM cartridges is that fonts stored in them cannot be changed. Any given cartridge may contain only a couple of desired fonts, which severely limits the flexibility of laser technology.

Flash memory could greatly enhance the speed and flexibility of laser printers. Using a flash-memory-based cartridge, users or work groups could develop their own font libraries. Since flash memory is nonvolatile, these fonts will stay resident in the printer without the use of batteries or uninterruptible power supplies.

Flash technology is well suited to application storage as well. Software stored on disk greatly reduces system performance, since disk access times are very slow compared with memory and CPU speeds (see figure). Of course, this speed bottleneck is not critical to most PC users; if a program takes a few seconds to load, so be it. For high-end systems, however, flash memory can boost system speed significantly by functioning as a code accelerator by storing programs or code that are accessed most often.

One high-end application that could benefit greatly from flash memory is CAD. Some CAD programs minimize the RAM used for program storage in order to maximize the data-memory space. To do this, the complex software swaps submodules into memory from disk as needed. Accessing the disk for a new module or library ties up the system and degrades performance.

Many CAD users install large add-in memory boards set up as RAM disks to avoid the transfer bottleneck. Whenever the system resets or powers down, however, this RAM disk loses its memory. A system containing a flash-memory disk emulator would not have this problem. Such a system for CAD and engineering applications is available from Digipro (Huntsville, Ala.).

■ PORTABLE PCs

Laptop and notebook-sized portable computers stand to benefit greatly from flash-memory technology. Although small, relatively efficient 2.5-inch disk

FOR HIGH-END SYSTEMS, FLASH MEMORY CAN BOOST SPEED BY FUNCTIONING AS A CODE ACCELERATOR

drives have come onto the market for laptop computers, their efficiency rating is derived primarily from power-management schemes. If no access has been made to the disk for a certain period of time, the

drive goes into a low-power standby mode. Unfortunately, the delay caused by switching from standby back to operational mode further impedes the already-slow disk-to-memory transfer.

From a power perspective, these small drives use relatively less power than desktop disk drives. Active power specifications in the 1-W range are attainable, as opposed to the multiple watts consumed by desktop-system drives. Compared with flash memories, which use about 50 mW during active reads, the 2.5-inch drives are power hungry. In addition, in standby mode flash memories use only 150 μW of power.

Notebook-sized computers do not have space for even the smallest of drives. Additionally, users of notebook-sized systems demand much longer battery operation than is typical of laptops. Typical battery operation for notebooks range from 50 to 100 hours, compared with two to five hours for laptops.

The typical notebook computer contains ROM cards for applications and SRAM for storage of data files. This architecture has a few flaws. Users who purchase software for a desktop system may not want to spend another few hundred dollars for ROM-card versions. If flash-memory storage is provided, users can download software at a much lower cost.

Some notebook-sized computers let users load applications into SRAM or low-refresh DRAM (pseudo-SRAM). This procedure, however, is highly dependent on the system's battery; when the battery dies, memory is erased. With flash memory, no power is consumed when the system is off. Additionally, flash memory is less expensive for bulk storage than is SRAM.

Psion, a company based in the United Kingdom, recently introduced a notebook computer based on flash memory. Psion compared the price differences between SRAM and flash memory and found flash memory to be much more cost-effective. Based on the power savings of flash memory and very tight system design criteria, Psion built a 60-hour operational system.

■ FILE STORAGE UNDER DOS

Given the compelling reasons to adopt flash memory in the various PC environments, designers can then ponder the question: How can a bulk-erasable memory be used for file storage under DOS? The DOS directory and file-allocation tables (FATs) require the ability to erase and rewrite single bytes and files. The answer to this has many facets and depends on the degree of flexibility needed.

The most inflexible, but easiest to implement, approach involves creating a fixed disk image. Before programming the flash memory, the vendor combines the operating system, utilities, and specific applications onto a disk. It then runs this suite of programs through a utility that adapts the software for ROM or flash-memory storage. Finally, another utility creates the DOS directory and FATs and assembles the disk image. Digital Research offers both a ROM-format DOS version and disk-image utilities. Microsoft offers an optimized ROM-format DOS for the portable market as well.

SOFTWARE EMBEDDED IN FLASH MEMORY CAN BE CHANGED WITHOUT COMPROMISING SYSTEM PERFORMANCE

Makers of notebook-sized computers, intelligent terminals, and dedicated handheld computers might consider this strategy. A laptop vendor that offers a low-power disk might consider this approach as well. The most-used software, such as the operating system, calendar, alarm, and communications software, will load more quickly and not burn as much power on each access. Upgrade-software disks could be sold to registered system owners already formatted with a new disk image.

Another approach currently used by Digipro involves adapting RAM-disk emulator drivers to flash memory. This entails trapping disk writes and programming the information algorithmically.

From the user perspective, the flash-memory disk operates as a hard disk, but with 100 times the typical read performance of a hard disk. To load the flash-memory disk emulator, a user simply issues the DOS Copy command.

A drawback to this approach is that loading any application or its libraries forces a rewrite of the entire directory and FATs. For example, a file that contains only 2 Kbytes and should program in 50 ms may take several seconds to store because of file-system overhead. Note that this drawback may not be important as far as code acceleration is concerned. Files are loaded to the flash-memory disk as an off-line task, so the write-performance impact is minimal to read-mostly performance.

A third solution to the DOS compatibility problem comes from Microsoft, which has developed a flash-memory file system that loads under MS-DOS and other compatible operating systems. When a user adds a directory or a file to the flash-memory disk, the file system adds the information in a linked list. Should the user delete the directory or file, the file system marks an attribute field as being inactive. When the disk fills up, the user copies the active files and directories to a hard disk on the desktop system or to another disk on a portable system. Since the DOS Copy command only grabs active files, the newly transferred version contains a clean, unfragmented linked list. The user then erases the original flash-memory disk.

As these examples illustrate, flash memory offers alternatives to current problems in system architecture. To offer true solutions, a memory technology must also satisfy three objectives: It must provide acceptable density, incur a reasonable cost, and offer EPROM-level reliability.

Intel's ETOX flash-memory technology is the first new technology in nearly 20 years to satisfy all three objectives. It is dense—the 1-Mbit 28F010 has been shipping in production units since April 1989. Costs have decreased dramatically as volume has ramped. And because the ETOX process and memory cell so closely parallel mainstream EPROM technology, flash memory has reliably doubled in density three times in the last two years. Additionally, a flash-memory device can be reprogrammed about 100,000 times—a sufficient number for most firmware and disk applications.

Current Intel flash-memory product offerings include the 28F256, 28F512, and 28F010. These devices are 32-Kbit × 8, 64-Kbit × 8, and 128-Kbit × 8, respectively. All products ship in either surface-mounted ceramic DIP or PLCC. Higher densities, plastic DIPs, and smaller packages will be available shortly. Additionally, for those designs requiring a bulk-memory solution similar to that offered by DRAM vendors, Intel offers eight 28F010 PLCC units mounted on a single in-line memory module (SIMM).

With flash-memory technology, software can be changed without compromising the performance of disk access. The task can be accomplished in a reliable manner and without the high costs associated with service calls by field technicians. Finally, flash memory is nonvolatile—data stay resident even when the power supply is cut off. Together, these attributes enable flash-memory technology to provide users with flexible firmware and improved system design. ■

ABOUT THE AUTHOR

SAUL ZALES *is a senior applications engineer at Intel Corp. in Folsom, Calif. He holds a BSEE from the University of Pennsylvania.*

Designing with Flash Memory

For many embedded-control applications, whether the control is of the sequential, closed-loop, or data-control variety, flash memory offers dense, reliable, rewritable, nonvolatile storage. From a systems perspective, the memory reads the same way as an EPROM, EEPROM, or SRAM, with access speeds of up to 120 ns. Intel fabricates flash-memory devices using its EPROM Tunnel Oxide (ETOX) process. ETOX is based on Intel's high-volume CMOS EPROM process. Because of this, flash memory reprograms in the same way as an EPROM—via CPU-controlled algorithms.

With flash memory, the designer implements the reprogramming algorithms, which are simple closed-loop algorithms that require 500 to 1,000 bytes of code. (Intel offers sample code generated for different base processors to minimize the software effort.) Because flash memory is bulk-erasable, this code must be stored and executed from another memory while the main code is updated. For this process, many designs rely on internal ROM space on microcontrollers or small EPROM boot loaders on Intel 80X86 systems. The boot loader contains sufficient code to initialize the system and reprogram the flash memory.

As with EPROM, flash memory requires a programming power supply of 12 V ±5 percent. Some systems feature a 12-V supply, but others do not. Systems with analog circuitry, for instance, often contain rails of 15 V or higher. In these systems, the programming power supply can be generated by regulating the higher voltage using an LM317-type regulator.

For memories that have a power supply of 5 V, designers can opt for either monolithic or discrete charge pumps, such as those as offered by Valor Electronics and Linear Technologies. Since flash memory requires only 30 mA per device, from the programming power supply during active programming and erasure, these boost circuits can be made fairly small.

Some flash-memory manufacturers claim to have simplified programming and erasure by developing automatic controls and 5-V-only programming. The 5-V technologies are based on EEPROM-programming rather than EPROM-programming techniques and generate high voltages internally. They require larger board spaces and are less dense, more costly, and less reliable than 12-V designs. And although they are called flash memories, these parts tend to be nothing more than EEPROM technologies.